Available online at www.sciencedirect.com

SCIENCE DIRECT®

Applied Mathematical Modelling 29 (2005) 263–276

APPLIED MATHEMATICAL MODELLING

www.elsevier.com/locate/apm

Yard crane scheduling in port container terminals

W.C. Ng *, K.L. Mak

Department of Industrial and Manufacturing Systems Engineering, The University of Hong Kong, Pokfulam Road, Hong Kong

Received 1 September 2003; received in revised form 1 August 2004; accepted 6 September 2004

Abstract

Yard cranes are the most popular container handling equipment for loading containers onto or unloading containers from trucks in container yards of land scarce port container terminals. However, such equipment is bulky, and very often generates bottlenecks in the container flow in a terminal because of their slow operations. Hence, it is essential to develop good yard crane work schedules to ensure a high terminal throughput. This paper studies the problem of scheduling a yard crane to perform a given set of loading/unloading jobs with different ready times. The objective is to minimize the sum of job waiting times. A branch and bound algorithm is proposed to solve the scheduling problem optimally. Efficient and effective algorithms are proposed to find lower bounds and upper bounds. The performance of the proposed branch and bound algorithm is evaluated by a set of test problems generated based on real life data. The results show that the algorithm can find the optimal sequence for most problems of realistic sizes.

Keywords: Yard crane scheduling; Container terminal; Branch and bound algorithm

1. Introduction

The globalization of trade and the subsequent breakdown of trade barriers have spurred tremendous growth in marine container traffic. Mega port container terminals, especially those in

* Corresponding author. Fax: +852 2858 6535.
E-mail address: ngwc@hku.hk (W.C. Ng).

0307-904X/$ - see front matter
doi:10.1016/j.apm.2004.09.009

Asia, have been working at, or close to, capacity. In addition, the rising competition among ports has put considerable pressure on them to improve their customer service. Among all the common service performance measures, the terminal turnaround time, i.e., the average period of time that a vessel stays in a terminal, is the key one. In most terminals, a large portion of the terminal turnaround time is spent on discharging and loading containers for a vessel.

In a port container terminal, a number of vessels are often berthed alongside, and each vessel is served by multiple quay cranes which are supported by a large number of yard cranes in the yard. Fig. 1 depicts the typical flow of containers in a container terminal. When a vessel arrives at the terminal, containers are normally discharged from the vessel, mounted onto trucks by quay cranes, and then unloaded by yard cranes at various locations in the yard for storage. In the loading operation, export containers loaded onto trucks by yard cranes at the yard are off-loaded at the quay and loaded onto a vessel by quay cranes. In both discharging and loading operations, the dispatching and scheduling of equipment are key planning problems faced by terminal planners.

A few days before a vessel arrival, terminal planners determine the sizes and locations of storage areas for the containers to be loaded onto the vessel, and then determine the containers to be stacked in each storage area in each day. A few hours before a vessel arrives, they determine the sequence of discharging and loading containers for the vessel and the quay cranes to be assigned to the vessel. Once handling operations start on the vessel, handling jobs of the quay cranes will generate transportation jobs for the trucks and handling jobs for the yard cranes. The transportation jobs are dispatched to highly mobile trucks on a real-time basis through voice communication or a wireless computer network. On the other hand, the movement of each of the less mobile yard cranes running on rubber tires is normally restricted to a predetermined zone within a container yard for a few hours before the crane moves to another zone, to ensure yard traffic safety and to avoid long zone-to-zone travelling time. Within a zone, a yard crane can move freely to perform all handling jobs generated by different vessels. The physical handling rate of a yard crane is generally about half of that of a quay crane, and very often, the container flow in a terminal is bottlenecked because of the slow yard crane operations. Hence, a good yard crane schedule can increase a terminal's throughput by increasing the container flow to and from vessels as a result of reducing the truck waiting time.

Fig. 2 shows a typical partial container yard layout of a container terminal: the yard is divided into multiple blocks called yard blocks; each yard block consists of a contiguous stretch of slots (40–60 slots); and each slot has several rows (6–8 rows). Each ground slot, denoted as a rectangle in the diagram, can store 5–7 containers. In most container terminals, zones are normally formed

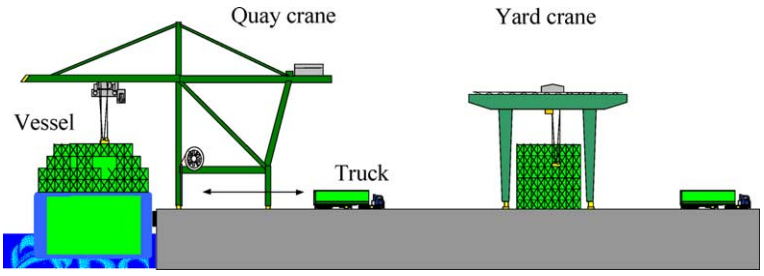

Fig. 1. Typical flow of containers in terminal operations.

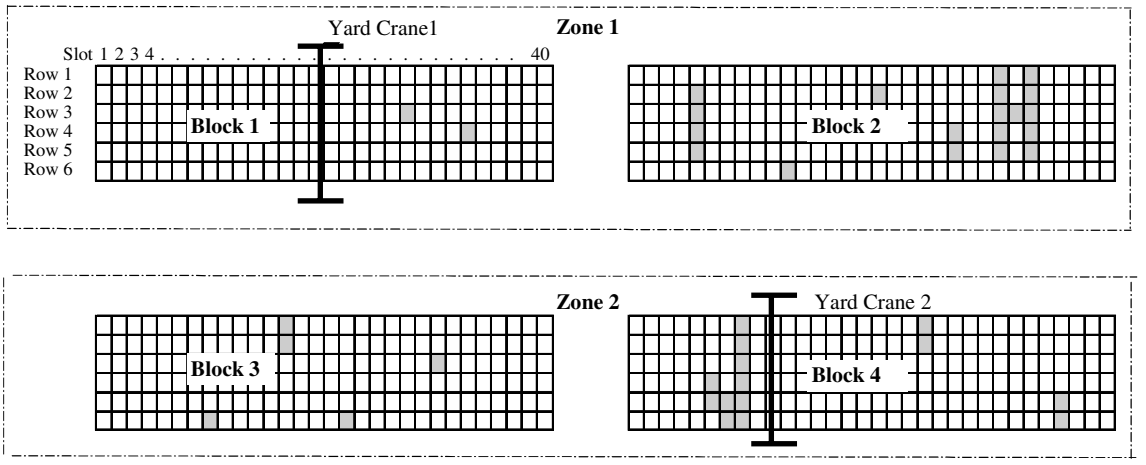

Fig. 2. Typical container yard layout of a container terminal.

by grouping adjacent yard blocks together so as to simplify the control of yard crane movements and to reduce the amount of time in which the yard cranes occupy truck travelling lanes. In the layout depicted in Fig. 2, there are two zones in the yard, zones 1 and 2, formed by grouping blocks 1 and 2, and blocks 3 and 4 together, respectively. The location of each job to be handled by a yard crane, as specified in the sequence of discharging and loading vessels is marked as a shaded rectangle in the diagram.

In mega port container terminals like those in Singapore and Hong Kong, it is important to reduce the waiting time of trucks by coordinating yard crane and truck movements. A terminal operations control system would normally determine the expected arrival times of the trucks for transporting containers from or to each yard block during the coming hour. Currently, human judgment is used to determine a schedule for each yard crane to perform handling jobs for the arriving trucks. The scheduling objective is to reduce the amount of the trucks' waiting time for the cranes.

Issues unique to port container terminal operations are receiving more attention due to increasing importance of marine transportation systems. The most important objective for a port container terminal is to increase its throughput or, in particular, to decrease the turnaround times of ships (see [9]). The turnaround time of a ship depends on the effectiveness of allocating and scheduling key resources, such as berths, yards, quay cranes, yard cranes and trucks.

Problems of scheduling resources, like vehicles and material handling equipment, arise frequently in logistics systems and these problems have been extensively studied under different settings (see [2]). Unfortunately, the results reported in most research literature are not directly applicable to a port container terminal due to its unique characteristics.

Research that specifically focuses on scheduling of quay cranes has been conducted by various researchers. Peterkofsky and Daganzo [10] and Daganzo [4] studied the static quay crane scheduling problem with the objective of minimizing the aggregate vessel delay cost and they proposed various heuristics and exact algorithms to solve the quay crane scheduling problem. Daganzo [5] studied the impact of quay crane scheduling strategies on terminal throughput and ship delay.

Several researchers have focused on the operational level issues particular to container loading or discharging operations of vessels, such as dispatching trucks and determining storage locations

for containers. Bish et al. [1] studied the problem of assigning each discharging container to a yard location and assigning dispatching vehicles to the containers so as to minimize the time to discharge all the containers from a ship. Kim et al. [7] developed a dynamic programming method to determine the storage location of export containers to minimize the number of relocation movements expected for the loading operation.

Recently, deployment and scheduling of container handling equipment in a container yard have been investigated by a number of researchers. Zhang et al. [11] studied the yard crane deployment problem with the objective of finding the times and routes of yard crane movements among yard blocks to minimize the total delayed workload in a container yard. A mixed integer programming model was proposed to determine the number of yard cranes to be deployed in each yard block in each planning period. A Lagrangian relaxation-based algorithm was used to obtain the optimal solution. Cheung et al. [3] extended the Zhang's model by relaxing the restriction that crane movement must be completed within a period. Kim and Kim [6] studied the problem of routing a single yard crane to support the loading operation of a vessel. A mixed integer program was proposed to determine the number of container pickups at each location and the sequence of locations to be visited by the yard crane, and an efficient optimal algorithm was developed. However, the assumption of having a dedicated group of yard cranes just to support vessels' loading operations is not a realistic one for port container terminals with a large number of berths.

In this paper, the problem of scheduling a yard crane to perform handling jobs with different ready times within its movement zone to minimize the sum of job waiting times is analyzed in detail. A mixed integer programming model is proposed in Section 2. In Sections 3–5, the properties of the problem are studied thoroughly and a branch and bound algorithm with efficient and effective bounding procedures is proposed on the basis of such properties. A numerical example is solved to illustrate the algorithm. Section 6 presents the results of computational experiments and the last section concludes the paper.

2. Mathematical model

In this section, the problem of scheduling a yard crane to handle all the jobs with different ready times within its movement zone is formulated as an integer program. There are n jobs in the zone to be handled by the yard crane in the current planning period. Let r_i, $i = 1, 2, \ldots, n$, be the ready time (truck arrival time) of job i, h_i, $i = 1, 2, \ldots, n$, be the time required by the yard crane to handle job i, d_{ij}, $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$, be the time required for the yard crane to travel from the location of job i to the location of job j, and d_{0j} be the time required for the yard crane to travel from the initial location of the yard crane to the location of job j. The n jobs are indexed in such a way that $r_i \leqslant r_{i+1}$. The following decision variables are needed to formulate the yard crane scheduling problem mathematically:

t_i the time at which the yard crane completes handling of job i,

$$X_{ij} = \begin{cases} 1 & \text{if job } i \text{ is handled before job } j, \\ 0 & \text{otherwise.} \end{cases}$$

Denote the set of t_i by $\boldsymbol{T}$ and the set of X_{ij} by $\boldsymbol{X}$. For the truck dispatched to job i, its waiting time in the yard is given by $t_i - h_i - r_i$. With given r_i, the problem of finding the optimal schedule that minimizes the sum of job waiting times can be stated as

Problem YCS:

$$\text{Minimize} \quad \sum_{i=1}^{n}(t_i - h_i - r_i)$$

subject to

$$t_i \geqslant r_i + h_i, \quad i = 1, 2, , \ldots, n, \tag{1}$$

$$t_j - t_i \geqslant d_{ij} + h_j - (1 - X_{ij})M, \quad i, j = 1, 2, \ldots, n, \text{ and } i \neq j, \tag{2}$$

$$X_{ij} + X_{ji} = 1, \quad i, j = 1, 2, \ldots, n, \text{ and } i \neq j, \tag{3}$$

$$X_{ij} \in \{0, 1\}, \quad i, j = 1, 2, \ldots, n, \text{ and } i \neq j. \tag{4}$$

The objective of Problem YCS is to minimize the sum of job waiting times. Constraints (1) give the relationship between a job's completion time, ready time and handling time. Constraints (2) give the relationship between the completion time of a job and that of its successors. Constraints (3) ensure the correctness of the precedence relationship specified by $\boldsymbol{X}$. Constraints (4) are simple binary constraints. As the set of t_i, $i = 1, 2, \ldots, n$, that minimizes $\sum_{i=1}^{n} t_i$, the total completion time, would also minimize $\sum_{i=1}^{n}(t_i - h_i - r_i)$, Problem YCS can be simplified as minimizing $\sum_{i=1}^{n} t_i$ subject to constraints (1)–(4).

Denote the job handled in the ith position of an n-job sequence by $[i]$ and the completion time of the job by $t_{[i]}$. For the case of $r_i = 0$ for $i = 1, 2, \ldots, n$, it can be shown that

$$t_{[i]} = \sum_{j=1}^{i}(h_{[j]} + d_{[j-1][j]}),$$

and that the total completion time is equal to

$$\sum_{i=1}^{n}\sum_{j=1}^{i}(h_{[j]} + d_{[j-1][j]}) = \sum_{i=1}^{n}(n - i + 1)(h_{[i]} + d_{[i-1][i]}).$$

It is noted that for this case, Problem YCS is a problem of non-preemptive scheduling with different job ready times on a single machine to minimize total completion time and the scheduling problem is an NP-complete problem (see [8]). A branch and bound algorithm is proposed in this paper to solve Problem YCS optimally. In the following section, the procedure for computing lower bounds is discussed in detail.

3. Lower bound

It can be shown from the definitions of $t_{[i]}$, $h_{[i]}$, $r_{[i]}$ and $d_{[i\text{-}1][i]}$ that for $i = 1, 2, \ldots, n$,

$$t_{[i]} = h[i] + \max\{t_{[i-1]} + d_{[i-1][i]}, r_{[i]}\}, \tag{5}$$

where $t_{[0]}$ is defined to be 0. It can be shown from the above equation that

$$t_{[i]} \geqslant \max\left\{\sum_{j=1}^{i}(h_{[j]} + d_{[j-1][j]}), r_{[i]} + h_{[i]}\right\}. \tag{6}$$

The above expression can be used to find a lower bound for $t_{[i]}$. In stating the lower bound, some extra notations are needed. Let $\alpha_i = \min\{h_j + d_{ij} | j = 1, 2, \ldots, n\}$, $\beta_i = \min\{h_i + d_{ji} | j = 1, 2, \ldots, n\}$, $\gamma_i = r_i + h_i$, and $\alpha_{[i]}$, $\beta_{[i]}$ and $\gamma_{[i]}$ be the ith smallest elements of the sets $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$, $\{\beta_1, \beta_2, \ldots, \beta_n\}$ and $\{\gamma_1, \gamma_2, \ldots, \gamma_n\}$, respectively. It can be deduced from the definitions of $\alpha_{[i]}$, $\beta_{[i]}$ and $\gamma_{[i]}$ and from expression (6) that there exists a lower bound for $t_{[i]}$, $\bar{t}_{[i]}$, satisfying the following expression:

$$\bar{t}_{[i]} = \max\left\{\alpha_0 + \sum_{j=1}^{i-1}\alpha_{[j]}, \sum_{j=1}^{i}\beta_{[j]}, \gamma_{[i]}\right\}, \tag{7}$$

where α_0 is defined to be $\min\{h_j + d_{0j} | j = 1, 2, \ldots, n\}$. It can be shown from the above expression that the total completion time of the n jobs satisfies the condition

$$\sum_{i=1}^{n} t_i \geqslant \sum_{i=1}^{n} \bar{t}_{[i]}. \tag{8}$$

It is obvious that expression (8) gives a lower bound for $\sum_{i=1}^{n} t_i$. In the following, the analysis for finding better lower bounds is outlined.

Let $\boldsymbol{S}_k = \{I_1, I_2, \ldots, I_k\}$ denote a partial sequence of length k, $k = 1, 2, \ldots, n$, where I_i, $i = 1, 2, \ldots, k$, denotes the ith job handled in the partial sequence. $\boldsymbol{S}_k$ is said to be feasible if I_i, $i = 1, 2, \ldots, k$, are all distinct. Furthermore, let $\boldsymbol{S}_k(j)$ be a feasible partial sequence of length k with job j as the last job in the partial sequence (i.e., $I_k = j$) and $g(\boldsymbol{S}_k(j))$ be the makespan of $\boldsymbol{S}_k(j)$. For a given $\boldsymbol{S}_k(j)$, consider the feasible partial sequence formed by appending job i to $\boldsymbol{S}_k(j)$. It can be deduced from constraints (1) and (2) and the definition of $\bar{t}_{[k+1]}$ that the following lemma is true.

Lemma 1. *$g(\{S_k(j), i\}) = \max\{r_i + h_i, g(S_k(j)) + d_{ji} + h_i\}$ where $g(\{S_k(j), i\})$ is the makespan of the feasible partial sequence formed by appending job i to $S_k(j)$.*

It can easily be shown from the definition of $\bar{t}_{[k+1]}$ and Lemma 1 that the following lemma is also true.

Lemma 2. *The minimum makespan of any feasible partial sequences of length $k + 1$ with job i as the last job is given by*

$$\max\left\{\bar{t}_{[k+1]}, r_i + h_i, \min_{\text{all feasible } \boldsymbol{S}_k(j)}\left[g(\boldsymbol{S}_k(j)) + d_{ji} + h_i\right]\right\}.$$

Let $\boldsymbol{\theta}_k(j)$ be the set containing m shortest feasible partial sequences of length k with job j as the last job. Hence, the makespan of a feasible partial sequence of length k with job j as the last job, not belonging to $\boldsymbol{\theta}_k(j)$, is greater than or equal to the makespan of $\boldsymbol{S}_k(j)$, $\boldsymbol{S}_k(j) \in \boldsymbol{\theta}_k(j)$. Thus, it is obvious that the following lemma is true.

Lemma 3. *If no feasible partial sequence can be formed by appending job i to $S_k(j)$ for all $S_k(j) \in \theta_k(j)$, $\max_{\boldsymbol{S}_k(j) \in \boldsymbol{\theta}_k(j)}\{g(\boldsymbol{S}_k(j)) + d_{ji} + h_i\}$ is a lower bound for the minimum makespan of*

any feasible partial sequences of length $k+1$ with job i as the last job and job j as the second last job.

Let $\lambda(i)$ be the set of job index j, $j = 1, 2, \ldots, n$, such that appending job i to each $\boldsymbol{S}_k(j) \in \boldsymbol{\theta}_k(j)$ results in infeasible partial sequences. By considering feasible partial sequences as well as infeasible partial sequences of length $k+1$ formed by appending job i to $\boldsymbol{S}_k(j)$, it can be shown using Lemmas 2 and 3 that Lemma 4 is true.

Lemma 4

$$\max \begin{cases} \bar{t}_{[k+1]} \\ r_i + h_i \\ \min \begin{cases} \min\limits_{j \notin \lambda(i)} \left[\min\limits_{\boldsymbol{S}_k(j) \in \boldsymbol{\theta}_k(j)} g(\boldsymbol{S}_k(j)) + d_{ji} + h_i \right] \\ \min\limits_{j \in \lambda(i)} \left[\max\limits_{\boldsymbol{S}_k(j) \in \boldsymbol{\theta}_k(j)} g(\boldsymbol{S}_k(j)) + d_{ji} + h_i \right] \end{cases} \end{cases}$$

is a lower bound for the minimum makespan of any feasible partial sequences of length $k+1$ with job i as the last job.

Denote the lower bound for the makespan of any feasible partial sequence of length k with job j as the last job in the sequence (i.e., $I_k = j$) by $\bar{g}_k(j)$. It is clear from the definition of $\bar{g}_k(j)$ that for $j = 1, 2, \ldots, n$, $\boldsymbol{S}_k(j) \geqslant \bar{g}_k(j)$ and hence, it can be shown from Lemma 4 that the following lemma is true.

Lemma 5

$$\bar{g}_{k+1}(i) = \max \left\{ \bar{t}_{[k+1]}, r_i + h_i, \min_{j \neq i} \left[\bar{g}_k(j) + d_{ji} + h_i \right] \right\}.$$

The above lemmas give useful insights for developing an algorithm to find a lower bound for any feasible partial sequence of length k, $k = 1, 2, \ldots, n$. It follows from Lemma 5 that $\bar{g}_1(i) = \max\left\{\bar{t}_{[1]}, r_i + h_i, d_{0i} + h_i\right\}$ is a lower bound for the makespan of any feasible partial sequences of length 1 with job i as the last job in the sequences. For $k = 2, 3, \ldots, n$, the following procedure is proposed for finding $\bar{g}_k(i)$, $i = 1, 2, \ldots, n$, a lower bound for the makespan of any partial sequences of length k with job i as the last job.

Procedure LB

Step 0. Set the lower bound for the makespan of the partial feasible sequence of length 1 with job i as the last job in the sequence, $\bar{g}_1(i)$, to $\max\left\{\bar{t}_{[1]}, r_i + h_i, d_{0i} + h_i\right\}$ for $i = 1, 2, \ldots, n$.
Initialize the set of the feasible partial sequence of length 1 with job i as the last job in the sequence, $\boldsymbol{\theta}_1(i) = \{i\}$ for $i = 1, 2, \ldots, n$.
Initialize the length of partial sequences to be considered by setting $\omega = 2$.
Step 1. Set job index $i = 1$.
Step 2. Set $j = 1$, $\boldsymbol{R} = \boldsymbol{Q} =$ null set, and $\min = \infty$.

Step 3. If $i \neq j$ then
if $\boldsymbol{\theta}_{\omega-1}(j) =$ null set then
if $\max\{\bar{t}_{[\omega]}, r_i + h_i, \bar{g}_{\omega-1}(j) + d_{ji} + h_i\} <$ min then set min $= \max\{\bar{t}_{[\omega]}, r_i + h_i, \bar{g}_{\omega-1}(j) + d_{ji} + h_i\}$;
else,
for each partial sequence $\boldsymbol{S}_{\omega-1}(j) \in \boldsymbol{\theta}_{\omega-1}(j)$, if the partial sequence of length ω formed by appending i to $\boldsymbol{S}_{\omega-1}(j)$, $\{\boldsymbol{S}_{\omega-1}(j), i\}$, is feasible, i.e. i and all the job indexes in $\boldsymbol{S}_{\omega-1}(j)$ are distinct, then
$\boldsymbol{Q} = \boldsymbol{Q} \cup$ the feasible partial sequence of length ω formed; set the makespan of the feasible partial sequence formed $= \max\{\bar{t}_{[\omega]}, r_i + h_i, [g(\boldsymbol{S}_{\omega-1}(j)) + d_{ji} + h_i]\}$
else,
$\boldsymbol{R} = \boldsymbol{R} \cup$ the infeasible partial sequence of length ω formed;
else,
goto Step 5.

Step 4. If $\boldsymbol{Q}$ is empty then
if the maximum makespan of the partial sequences in $\boldsymbol{R}<$min then set min = the maximum makespan of the partial sequences in $\boldsymbol{R}$.

Step 5. Set $j = j + 1$. If $j \leqslant n$ goto Step 3.

Step 6. If $\boldsymbol{Q}$ is non-empty, i.e. there exist some feasible partial sequences of length ω with job i as the last job, then
$\boldsymbol{\theta}_{\omega}(i) =$ the set of the first m elements in $\boldsymbol{Q}$ ranked in ascending order of makespan with their makespans $<$ min, set $\bar{g}_{\omega}(i) =$ the minimum makespan of the feasible partial sequences in $\boldsymbol{\theta}_{\omega}(i)$;
else,
set $\bar{g}_{\omega}(i) = \max\{r_i + h_i, \bar{t}_{[\omega]}, \text{min}\}$.

Step 7. Set $i = i + 1$ to consider the next job;
if $i \leqslant n$ goto Step 2.

Step 8. Set $\omega = \omega + 1$.
if $\omega \leqslant k$ then goto Step 1; else, terminate.

In the above procedure, Step 0 determines a lower bound for the makespan of the feasible sequence of length 1 with job i as the last job using the result of Lemma 5. For each i, $i = 1, 2, \ldots, n$, and ω, $\omega = 2, 3, \ldots, k$, by considering all possible j as the $(\omega - 1)$th job, Steps 2–7 finds $\boldsymbol{\theta}_{\omega}(i)$, the set containing feasible partial sequences of length ω with job i as the last job and with makespan shorter than the shortest makespan of infeasible partial sequences, if they exist, and set the lower bound equal to the shortest makespan of all the feasible partial sequences in $\boldsymbol{\theta}_{\omega}(i)$. Otherwise, the lower bound is set to the value of min found in Steps 3 and 4. Then, Step 6 determines $\bar{g}_{\omega}(i)$ based on the results of Lemmas 4 and 5.

It is noted that the n-job sequence determined by Procedure LB possesses the interesting property given in the following lemma.

Lemma 6. *Let i_k be the i that minimizes $\bar{g}_k(i)$, $k = 1, 2, \ldots, n$, and $\boldsymbol{S}_n^* = \{I_1^*, I_2^*, \ldots, I_n^*\}$ be the sequence determined in Procedure LB that minimizes $\bar{g}_n(i_n)$. If $\boldsymbol{S}_n^*$ is feasible (i.e. I_k^* are all distinct) and $I_k^* = i_k$ for $k = 1, 2, \ldots, n$ then $\boldsymbol{S}_n^*$ is the optimal sequence for Problem YCS.*

Proof. If I_k^* are all distinct, X_{ij}, implied by I_k^* found, and $\bar{g}_k(i_k)$, determined in Steps 3 and 6 of Procedure LB, are feasible to constraints (1)–(4). Let $\{\hat{T}, \hat{\mathbf{X}}\}$ be the optimal solution of Problem YCS and $\hat{Y}_{ki}$ be a parameter assuming value of 1 if job i is the kth job handled in the optimal sequence. It follows from the definition of i_k, $\bar{g}_k(i)$, $\hat{t}_i$ and $\hat{Y}_{ki}$ that $\sum_{i=1}^{n} \hat{Y}_{ki}\hat{t}_i \geqslant \bar{g}_k(i_k)$ for all k. Thus, if $I_k^* = i_k$ for $k = 1, 2, \ldots, n$, $\{X_{ij}, \bar{g}_k(i_k)\}$ is the optimal solution of Problem YCS.

If $\boldsymbol{S}_n^*$ determined in the lower bounding procedure does not possess the property stated in the above lemma, $\bar{g}_k(i)$ can still be used to find a lower bound for $\sum_{i=1}^{n} t_i$. It is obvious from the definition of $\bar{g}_k(i)$ that if $\hat{Y}_{ki} = 1$ then $\hat{t}_i \geqslant \bar{g}_k(i)$. Hence,

$$\sum_{i=1}^{n}\sum_{k=1}^{n} \hat{Y}_{ki}\hat{t}_i \geqslant \sum_{i=1}^{n}\sum_{k=1}^{n} \hat{Y}_{ki}\bar{g}_k(i). \tag{9}$$

However, $\sum_{i=1}^{n}\sum_{k=1}^{n} \hat{Y}_{ki}\bar{g}_k(i)$ cannot be used directly as a lower bound for $\sum_{i=1}^{n} t_i$ as one would not know $\hat{Y}_{ki}$ without determining the optimal solution of Problem YCS. In the following, a method is proposed to determine a lower bound for $\sum_{i=1}^{n}\sum_{k=1}^{n} \hat{Y}_{ki}\bar{g}_k(i)$.

Let Z_{ki} be a binary variable assuming value of 1 if job i is the kth job handled in a sequence. Consider the following optimization problem:

$$\text{Minimize} \quad \sum_{i=1}^{n}\sum_{k=1}^{n} Z_{ki}\bar{g}_k(i)$$

subject to

$$\sum_{k=1}^{n} Z_{ki} = 1, \quad i = 1, 2, \ldots, n, \tag{10}$$

$$\sum_{k=1}^{n} Z_{ki} = 1, \quad k = 1, 2, \ldots, n, \tag{11}$$

$$Z_{ki} \in 0, 1, \quad i, k = 1, 2, \ldots, n. \tag{12}$$

The above optimization problem is a simple assignment problem which can easily be solved. Let $\hat{Z}_{ki}$, $i, k = 1, 2, \ldots, n$, be the optimal solution of the above problem. It is obvious form the definition of $\hat{Y}_{ki}$ that $\hat{Y}_{ki}$, $i, k = 1, 2, \ldots, n$, satisfies constraints (10)–(12) and hence is a feasible solution of the above assignment problem. Hence, $\sum_{i=1}^{n}\sum_{k=1}^{n} w_i\hat{Y}_{ki}\bar{g}_k(i) \geqslant \sum_{i=1}^{n}\sum_{k=1}^{n} w_i\hat{Z}_{ki}\bar{g}_k(i)$ and thus, it is obvious from expression (9) that $\sum_{i=1}^{n}\sum_{k=1}^{n} w_i\hat{Y}_{ki}\hat{t}_i \geqslant \sum_{i=1}^{n}\sum_{k=1}^{n} w_i\hat{Z}_{ki}\bar{g}_k(i)$, which clearly indicates that $\sum_{i=1}^{n}\sum_{k=1}^{n} w_i\hat{Z}_{ki}\bar{g}_k(i)$ is a lower bound for the optimal objective value of Problem YCS.

A numerical example is used in the following to illustrate each step of the procedure for finding a lower bound for the optimal solution of Problem YCS. Consider the problem of scheduling a yard crane to perform 5 handling jobs with different job ready times: $h_i = 4$ for $i = 1, 2, \ldots, 5$, $r_1 = 2$, $r_2 = 5$, $r_3 = 7$, $r_4 = 13$, and $r_5 = 15$. The travelling time between locations of jobs i and j is: $d_{01} = 2$, $d_{02} = 5$, $d_{03} = 2$, $d_{04} = 1$, $d_{05} = 2$, $d_{12} = 3$, $d_{13} = 0$, $d_{14} = 1$, $d_{15} = 4$, $d_{23} = 3$, $d_{24} = 4$, $d_{25} = 7$, $d_{34} = 1$, $d_{35} = 4$, $d_{45} = 3$ and $d_{ij} = d_{ji}$.

With the above problem parameters, it can be determined from expression (7) that $\bar{t}_{[1]} = 6$, $\bar{t}_{[2]} = 9$, $\bar{t}_{[3]} = 13$, $\bar{t}_{[4]} = 19$ and $\bar{t}_{[5]} = 26$. The lower bound given by expression (8) is 73. Table 1 summarizes the working of Procedure LB for the case of $m = 1$.

Table 1
Summary of computations of Procedure LB for the example problem

k	i	$\bar{g}_k(i)$	$\boldsymbol{\theta}_k(i)$
1	1	6	{1}
	2	9	{2}
	3	11	{3}
	4	17	{4}
	5	19	{5}
2	1	15	{3, 1}
	2	13	{1, 2}
	3	11	{1, 3}
	4	17	{1, 4}
	5	19	{1, 5}
3	1	15	Null set
	2	18	{1, 3, 2}
	3	19	Null set
	4	17	{1, 3, 4}
	5	19	{1, 3, 5}
4	1	22	Null set
	2	22	Null set
	3	19	Null set
	4	20	Null set
	5	23	Null set
5	1	26	Null set
	2	26	Null set
	3	26	Null set
	4	26	Null set
	5	27	Null set

Since $\boldsymbol{S}_n^*$ is infeasible, the optimality condition in Lemma 6 is not met and thus, one has to solve the assignment problem. It is found that the lower bound given by the optimal objective function value of the assignment problem, $\sum_{i=1}^{n}\sum_{k=1}^{n}\hat{Z}_{ki}\bar{g}_k(i)$, is 82.

4. Upper bound

To make a branch and bound algorithm more efficient, an efficient and effective heuristic is needed. The heuristic proposed in this section, which is a simple modification of Procedure LB, computes the amount of contribution of each job to the total completion time and then finds the partial sequence with the smallest sum of these contributions.

It can be shown from expression (5) that

$$t_{[i]} = \sum_{j=1}^{i} f_{[j]},$$

where $f_{[j]} = h_{[j]} + \max\{d_{[j\text{-}1][j]}, r_{[j]} - t_{[j\text{-}1]}\}$. Hence, the total completion time is given by

$$\sum_{i=1}^{n} t_{[i]} = \sum_{i=1}^{n} (n - i + 1) f_{[i]}.$$

Thus, in view of the above equation, $(n - i + 1) f_{[i]}$ can be used to measure the amount of contribution of the job in position i to the total completion time. Detailed steps of the upper bound heuristic are as follows:

Heuristic UB

Step 0. Initialize the set of unscheduled jobs $= \{1, 2, \ldots, n\}$ and $i = 1$.
Step 1. Find job i^*, the job in the set of unscheduled jobs with the smallest $(n - i + 1) f_{[i]}$, schedule job i^* as the ith job and delete the job from the set of unscheduled jobs. Repeat this step for $i = 2, 3, \ldots, n$.

The yard crane scheduling problem in Section 3 is again used to illustrate the steps of Heuristics UB. The sequence determined by Heuristic UB is 1–3–4–5–2 with a total completion time of 93.

5. Branch and bound algorithm

In this section, a branch and bound algorithm is developed, which uses the lower bound and the upper bound derived in the previous sections. The branching rule used is to select the job with the smallest lower bound in the set of jobs to be sequenced. In order to obtain a feasible solution quickly and to control the computer storage requirement, the search strategy used is depth first.

It is worthwhile to note that with some simple modifications, the analysis results presented in Section 3 can easily be extended to find a lower bound for the optimal objective value of Problem YCS with a given feasible partial sequence. For a given π_q $(q = 1, 2, \ldots, n)$, the sequence of the first q jobs, Procedure LB can easily be modified to find a lower bound for the sum of completion times from the $(q + 1)$th job to the nth job by replacing $\bar{g}_1(i) = \max\{\bar{t}_{[1]}, r_i + h_i, d_{0i} + h_i\}$ by $\bar{g}_{q+1}(i) = \max\{\bar{t}_{[q+1]}, r_i + h_i, d_{[q]i} + h_i\}$ and $k = 2$ by $k = q + 2$ in Step 0. Likewise, with some minor modifications on Lemma 6, the lemma can be used to test, for a given π_q, whether the optimal solution of the corresponding assignment problem gives the optimal sequence for the remaining $(n - q)$ jobs. If the optimal sequence is found, the node is fathomed.

The example problem in Section 3 is again used to illustrate the algorithm. The initial upper bound determined in Section 4 is 93. Using the lower bounding procedure in Section 3, the lower bound for the set of pending nodes in the tree can easily be found. Fig. 3 shows the solution tree of the problem.

In the above figure, the number above a node is the lower bound found by the lower bounding method described in Section 3 taking into account the scheduling decisions made up to that node, and an underlined number above a node indicates the case that the lower bound found is the same as the smallest total completion time attainable for the scheduling decisions made up to that node, i.e. the lower bounding method finds the optimal schedule along the branch leading to the node. The branch and bound algorithm stops when it finds that the lower bound of the branch 1–3–4

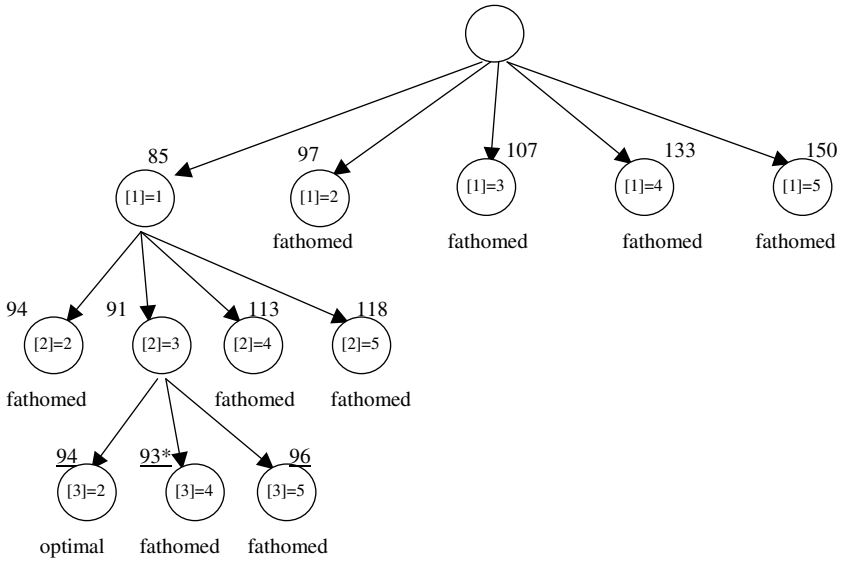

Fig. 3. The branching tree for the example problem.

found is the smallest and the lower bounding method finds the optimal solution for the branch. The optimal schedule found by the branch and bound algorithm is 1–3–4–5–2 which is the same as the one found by Heuristic UB.

6. Computational results

The branch and bound algorithm described in the previous section was implemented as a C++ program on a Pentium 200 MHz personal computer. Its performance was evaluated by solving a set of 40 test problems which were generated using realistic data. It was found from the yard-crane operational data collected from container terminals in Singapore and Hong Kong that h_i ranges from 2 to 4 minutes and d_{ij} from 0 to 10 minutes, and n from 0 to 20 jobs. For $n = 10$, 15, 20 and 25, 10 test problems with h_i and d_{ij} randomly selected from the typical ranges. The randomly generated problems were first solved by CPLEX and it was found that CPLEX failed to find the optimal schedule within 30 minutes for the problems with $n \geqslant 20$. The 40 randomly generated test

Table 2
The performance of the branch and bound algorithm

N	CPU (seconds)		
	Mean	Minimum	Maximum
10	0.17	0.04	0.30
15	1.33	0.79	1.72
20	18.20	3.37	62.41
25	444.72	50.30	1464.40

problems were then solved by the branch and bound algorithm. In running the branch and bound algorithm, m is set to 1 in Procedure LB and m is set to $2n$ in Heuristic UB.

Table 2 summarizes the performance of the algorithm.

As expected, the above results indicate that the required CPU time increases exponentially as n increases. It is noted from Table 2 that for most of the test problems ($n < 25$), the branch and bound algorithm can solve Problem YCS in a reasonable amount of time.

7. Conclusions

In this paper, the problem of scheduling a yard crane to handle jobs with different ready times within its movement zone has been studied. A mixed integer program has been proposed to solve the problem, and the properties of the problem have been studied in detail. On the basis of these properties, a branch and bound algorithm with efficient and effective bounding procedures has been developed. A numerical example has been used to illustrate the procedures of the algorithm. The performance of the algorithm has been evaluated by using a set of test problems. The computational results have shown that the algorithm works well for most of the test problems.

Acknowledgments

The authors would like to thank Dr. Leong of PSA Corporation, Singapore, and Dr. Lai of Hongkong International Terminals Limited for many useful discussions, comments and suggestions. The author would also like to thank anonymous referees for their valuable comments. This research was supported by the HKU CRCG grant and RGC Competitive Earmarked Research Grant (HKU 7193/04E).

References

[1] E.K. Bish, F. Chen, T. Leong, C. Li, J.W.C. Ng, D. Simchi-Levi, Analysis of a new scheduling and location problem, Naval Res. Logist. 46 (2001) 363–385.
[2] J. Bramel, D. Simchi-Levi, The Logic of Logistics: Theory, Algorithms, and Applications for Logistics Management, Springer-Verlag, New York, 1997.
[3] R.K. Cheung, C.L. Li, W. Lin, Interblock crane deployment in container terminals, Transport. Sci. 36 (2002) 79–93.
[4] C.F. Daganzo, The crane scheduling problem, Transport. Res. B 23 (1989) 159–175.
[5] C.F. Daganzo, Crane productivity and ship delay in ports, Transport. Res. Rec. 1251 (1990) 1–9.
[6] K.H. Kim, K.Y. Kim, An optimal routing algorithm for a transfer crane in port container terminals, Transport. Sci. 33 (1999) 17–33.
[7] K.H. Kim, Y.M. Park, K.R. Ryu, Deriving decision rules to locate export containers in container yards, Eur. J. Oper. Res. 124 (2000) 89–101.
[8] J.K. Lenstra, A.H.G. Rinnooy, P. Brucker, Complexity of machine scheduling problems, Ann. Discrete Math. 1 (1977) 343–362.
[9] McKinsey and Company, Inc., Containerization: the key to low transport, a report by McKinsey and Company, Inc. for the British Transport Docks Board, 1967.

[10] R.I. Peterkofsky, C.F. Daganzo, A branch and bound solution method for the crane scheduling problem, Transport. Res. B 24 (1990) 159–172.
[11] C. Zhang, Y. Wan, J. Liu, R.J. Linn, Dynamic crane deployment in container storage yards, Transport. Res. B 36 (2002) 537–555.